**CST-305: Project 2 – Runge-Kutta-Fehlberg (RKF) for ODE**

**Jordan Scott & Angel Velazquez**

**Responsibilities:**

Both members were focused on the program, whereas Jordan focused on doing the RKF method by hand and Angel verified the work by using the program and completing the README file. Both members also did the documentation and review the code.

**System Performance Context:**

The program runs and shows the first 5 solutions, then the program shows the first 1,000 to 2,000 solutions through a graph that the program plots the data on.

**Specific Problem:**

The specific problem solved is creating a Python program to solve an ODE for the function and doing the math by hand, afterward verifying the answer by using the ODE on the Python program and matching the results. Then the program is also supposed to show not just solutions for the first 5 solutions, but up to 1000 to 2000 depending on the hardware that the program is running on.

**Mathematical Approach:**

The approach uses the Runge-Kutta Fehlberg method. This method consists of the following.

You are given and h.

To find , we need to use where:

And

To find the rest of the values of x and y we would use the following:

**Approach for implementation in code:**

The approach for implementing this in Python was importing the formula of the ODE into Python code. We can do this by defining a function ODE that has parameters that take in a x and y value and have a variable dydx that shows the ODE function we are solving for . Afterwards, we need to define the initial x to equal 1and y to equal 5 and respectively along with the step-size(h) being 0.02.

I will finish the rest of this section once I can see the final code.

The ODE is . Calculate manually for . Populate the table below accordingly.

|  |  |  |  |
| --- | --- | --- | --- |
| Method: RUNGE-KUTTA METHOD | | | |
| Problem: | | | |
|  |  | | True Solution |
|  |  |
|  |  |  |  |
|  | 1.02 |  |  |
|  | 1.04 |  |  |
|  | 1.06 |  |  |
|  | 1.08 |  |  |
|  | 1.1 |  |  |

I will finish the table once I get the true solution and check the values of the y’s with the code.

**Screenshots:**

Graph of the RK4 solution:

Graph of the ODE solution:

Graph displaying the solutions together:

Both RK4 and ODE solutions together:

Over 1,000 points for ODE and RK4

Console displaying the elapsed time from the program:

**Github Repo:** https://github.com/angel-vlzqz/Modeling-and-Simulation/tree/main/projects/project%202%20CLC